Amendments to the Specification

Please replace the paragraph beginning on page 8, line 24 with the following amended paragraph:

Figure 6 shows an example input/output subsystem ~~peripheral controller 162~~;

Please replace the paragraph beginning on page 14, line 8 (as amended in paper no. 8) with the following amended paragraph:

3D graphics processor 154 performs graphics processing tasks. Audio digital signal processor 156 performs audio processing tasks. Display controller 164 accesses image information from main memory 112 and provides it to video encoder 120 for display on display device 56. Audio interface and mixer 1300 interfaces with audio codec ~~code~~ 122, and can also mix audio from different sources (e.g., streaming audio from mass access storage device 106, the output of audio DSP 156, and external audio input received via audio codec 122). Processor interface 150 provides a data and control interface between main processor 110 and graphics and audio processor 114.

Please replace the paragraph beginning on page 15, line 20 with the following amended paragraph:

Figure 5 is a logical flow diagram of graphics processor 154. Main processor 110 may store graphics command streams 210, display lists 212 and vertex arrays 214 in main memory 112, and pass pointers to command processor 200 via bus interface 150. The main processor 110 stores graphics commands in one or more graphics first-in-first-out (FIFO) buffers 210 it allocates in main memory 112 ~~110~~. The command processor 200 fetches:

- command streams from main memory 112 via an on-chip FIFO memory buffer 216 that receives and buffers the graphics commands for synchronization/flow control and load balancing,
- display lists 212 from main memory 112 via an on-chip call FIFO memory buffer 218, and

- vertex attributes from the command stream and/or from vertex arrays 214 in main memory 112 via a vertex cache 220.

Please replace the paragraph beginning on page 16, line 8 with the following amended paragraph:

Command processor 200 performs command processing operations 200a that convert attribute types to floating point format, and pass the resulting complete vertex polygon data to graphics pipeline 180 for rendering/rasterization. A programmable memory arbitration circuitry 131 ~~130~~ (see Figure 4) arbitrates access to shared main memory 112 between graphics pipeline 180, command processor 200 and display controller/video interface unit 164.

Please replace the paragraph beginning on page 16, line 20 with the following amended paragraph:

Transform unit 300 performs a variety of 2D and 3D transform and other operations 300a (see Figure 5). Transform unit 300 may include one or more matrix memories 300b for storing matrices used in transformation processing 300a. Transform unit 300 transforms incoming geometry per vertex from object space to screen space; and transforms incoming texture coordinates and computes projective texture coordinates (300c). Transform unit 300 may also perform polygon clipping/culling 300d. Lighting processing 300e also performed by transform unit 300 ~~300b~~ provides per vertex lighting computations for up to eight independent lights in one example embodiment. Transform unit 300 can also perform texture coordinate generation (300c) for embossed type bump mapping effects, as well as polygon clipping/culling operations (300d).

Please replace the paragraph beginning on page 34, (numbered) line 8 with the following amended paragraph:

Figure 8 is an even more detailed overall view of serial interface 1000 ~~showing the details of serial interface communication circuitry and registers 1012~~. Controllers 52a

and 52b (and 52c and 52d, if present) are connected to game console 54 via connector ports 1002. Modem 1404 modulates and demodulates data transferred between the controllers and the console. In the example system, communication between the console and the controllers uses duty-cycle (pulse-width) modulation and the data is communicated over one line. The communication is half-duplex. The byte transfer order is "big-endian" in which within a given multi-byte numeric representation, the most significant byte has the lowest address (i.e., the data is transferred "big-end" first). Controller input/output buffer 1016 is used for normal data transfers involving controllers 52a-52d. As shown in Figure 8 and as will be explained in greater detail below, input/output buffer 1016 is arranged as a double buffer. Communication RAM 1014 is provided for use in variable-size data transfers to and from controllers 52a-52d. In the example system, the maximum data size of these variable-size data transfers is 32 words. Of course, the present invention is not limited in this respect. Channel selector circuit 1408 controls selectors 1412a-1412d to selectively connect modem 1404 to either communication RAM 1014 or input/output buffer 1016. An HV counter latch circuit 1406 latches the screen position of a flash signal when a trigger input is received from a light gun unit. In the example system shown in Figure 8, triggers inputs to the HV counter latch circuit 1406 are provided for connectors 1 and 2 only. It will be apparent that trigger inputs may be provided for the other connectors if desired. HV counter latch circuit 1406 may also be used with light pens connected to connectors 1 and/or 2.

Please replace the paragraph beginning on page 39, line 23 with the following amended paragraph:

Ordinarily, output data is copied from buffer 1 to buffer 0 immediately after it is written to output buffer 1 by main processor 110. However, the copying may also be timed to start with vertical blanking in order to control the timing of commands to 3D LCD shutter glasses connected to display controller <u>164</u> ~~162~~. To enable such timing, VBCPY may be set to "1" so that copies to buffer 0 occur only upon vertical blanking. The default value for the VBCPY bits is "0".

Please replace the paragraph beginning on page 40, line 3 with the following amended paragraph:

The size of the data involved in the polling is fixed. However, there are occasions on which it is desirable to transfer larger amounts of data between main console 54 and controllers 52 (e.g., writing game data to a nonvolatile memory built-in or attached to controller 52). Example serial interface 1000 ~~100~~ provides the capability of transferring larger amounts of data in accordance with a process that is explained with reference to Figure 11. SICOMCSR (SI Communication Control Status Register) includes a register OUTLNGTH for setting the communication channel output length (in bytes); INLNGTH for setting the communication channel input length (in bytes); and CHANNEL for determining which serial interface channel will be used. The minimum transfer that may be designated by OUTLNGTH and INLNGTH is 1 byte. A value of 0x00 will transfer 128 bytes. To transfer data, command and output data is set to communication RAM 1014 ~~1410~~ and main processor 110 sets the OUTLNGTH, INLNGTH and CHANNEL registers. The setting ("1") of the TSTART bit register causes execution of the current communication transfer. The transfer begins immediately after the current transaction on the designated channel is completed. When read, the TSTART bit represents the current transfer status. Once a communication transfer is executed, polling resumes at the next vertical blanking interval if the channels EN bit is set.

Please replace the paragraph beginning on page 40, line 21 with the following amended paragraph:

When main processor 110 sets the TSTART bit to "1", the transfer mode of the channel designated in the CHANNEL register is changed to Communication RAM mode by selector circuit 1408 (see Figure 8) which controls the appropriate selector circuit to connect that channel to communication RAM 1014 ~~1410~~. In the example of Figure 11, channel 1 is the designated channel. Output data is then transmitted from communication RAM 1014 ~~1410~~ and is output to controller 52 via modem 1404. Controller 52 may also transmit input data to channel 1 and this data is communicated to communication RAM 1014 ~~1410~~ via selector 1412a. The input data is pushed from communication RAM 1014

~~1410~~ to main processor 110. Modem 1404 automatically clears the TSTART bit when the transfer is complete.

Please replace the paragraph beginning on page 41, line 5 with the following amended paragraph:

During Communication RAM transfer mode on channel 1, the remaining channels (i.e., channels 2, 3 and 4) continue to transfer data using input/output buffer 1016 ~~1414~~. During communication RAM transfers, main processor 110 generates an Addr CRC and transmits the CRC to the controller. The controller processes the Addr CRC to check that the data has been correctly received. In the case of a communication RAM transfer, the controller returns Data CRC. The main processor 110 generates a data CRC and compares this with the controller returned Data CRC.